An Algorithm for Diagnostic Fault Simulation

Yu Zhang and Vishwani D. Agrawal

Auburn University, Department of Electrical and Computer Engineering, Auburn, AL 36849, USA

yzz0009@auburn.edu, vagrawal@eng.auburn.edu

Abstract--In diagnostic testing faults detectable by test vectors are partitioned into groups. This partitioning is such that a fault is distinguishable from faults in all other groups, but is indistinguishable from those in its own group. Diagnostic fault coverage (DC) is defined as the number of fault groups divided by the total number of faults. We present a new diagnostic fault simulation algorithm that determines the DC of given test vectors and produces a fault dictionary. For each vector, we begin with detected fault list at each primary output obtained from a convetional fault simulator. For the vector being simulated each fault is assigned a detection index that uniquely specifies its detection status at all primary outputs. Fault list is then partitioned. Faults with different detection index are distinguished by the simulated vector and are kept in separate groups. Any fault in a group by itself is dropped from further simulation with subsequent vectors for which its detection index remains unknown (X). After simulation of each vector, the cumulative DC is obtained by counting the fault groups. Fault dictionary syndrome for a fault is the array of its detection indexes.

1 Introduction

A common objective of testing is to detect all or most modeled faults. Although fault coverage (percentage or fraction) has a somewhat nonlinear relationship with the tested product quality or defect level (parts per million) for practical reasons fault coverage continues to be a measure of the test quality [3]. Tests can be random, functional or algorithmically generated vecctors. However, a fault simulator is an essential tool for obtaining meaningful tests. A simulator determines how many faults have been detected by given vectors thus providing the fault coverage. Years of research has produced highly efficient fault simulation algorithms and programs [3].

Some modern test scenarios go beyond fault detection. Here we diagnose or identify the fault causing failure.

A 100% fault coverage (FC) in the traditional sense means that the tests detect all modeled faults. This does not mean that those tests will identify the fault. A recent paper defines a diagnostic coverage (DC) metric [16]. A 100% DC means that each modeled fault is distinguished from all others. A diagnostic fault simulation algorithm is the contribution of this paper. Key features of the new algorithm are (1) it accepts fault detection data from any conventional fault simulator thus benefitting from the efficiency of a matured program, (2) fault dropping is used to delete diagnosed faults from the list of faults as fault simulation progresses, and (3) for a given set of input vectors it provides fault coverage (FC), diagnostic coverage (DC), and necessary data for fault dictionary.

In [15], a diagnostic fault simulation method has been presented. In that method, during simulation, faults are grouped into classes. Faults with identical output responses are put in the same class. Fault pairs consisting of faults from the same class are sent to an equivalence identification tool. If the fault pair is proved equivalent, one fault is dropped from the fault list. By concurrently performing diagnostic fault simulation and equivalence identification, the simulation time is greatly reduced. In our method a fault is dropped when it is distinguished from all other faults; this is achieved without fault equivalence checking, though we can also benefit from it.

In [14], a diagnostic fault simulator is constructed for sequential circuits. Because of the possibility of the X state at a primary output in a sequential circuit, additional information has to be stored for diagnosis. For example, consider a sequential circuit with three primary outputs. For a certain input vector, suppose fault 1 has a response 1X0 and fault 2, X1X. These two faults are said to be *potentially distinguished*. However this is not the case for combinational or full scan circuits for which the simulation method described in this paper can be more memory and time efficient.

In [6], a diagnostic test pattern generator (DATPG) for combinational circuits is presented. It aims at generating distinguishing test vectors for given fault pairs and no diagnostic fault simulation process is used. Such DATPG can greatly benefit from a fast simulation scheme.

2 Diagnostic Coverage Metric

For a set of vectors we group faults such that all faults within a group are not distinguishable from each other by the given vectors, while each fault in a group is pair-wise distinguishable from every fault in any other group. This grouping is similar to equivalence collapsing except here the grouping is conditional to the vectors. If we generate a new vector that detects a subset of faults in a group then that group is partitioned into two groups, one containing the detected subset and the other containing the rest. Suppose, we have sufficient vectors to distinguish between every fault pair, then there will be as many groups as faults and every group will have just one fault. Prior to test generation all faults are in a single group we will call g_0. As tests are generated, detected faults leave g_0 and start forming new groups, $g_1, g_2, \ldots g_n$, where n is the number of distinguishable fault groups. For perfect detection tests g_0 will be a null set and for perfect diagnostic tests, $n = N$, where N is the total number of faults. *Diagnostic coverage, DC*, has been defined as [16],

$$DC = \frac{Number\ of\ detected\ fault\ groups}{Total\ number\ of\ faults} = \frac{n}{N} \quad (1)$$

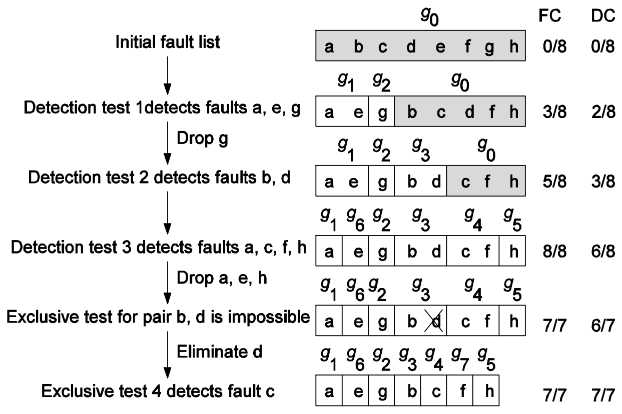
Initially, without any tests, $DC = 0$, and when all faults are detected and pair-wise distinguished, $DC = 1$. Also, the numerator in equation (1) is the number of fault dictionary syndromes [3] and the reciprocal of DC is the *diagnostic resolution* (DR) [1]. For completeness of this discussion, detection fault coverage (FC) is,

$$FC = \frac{Number\ of\ detected\ faults}{Total\ number\ of\ faults} = \frac{N - |g_0|}{N} \quad (2)$$

3 Diagnostic Fault Simulation Algorithm

We explain the simulation algorithm using a hypothetical example given in Figure 1. Suppose a circuit has eight faults ($N = 8$), identified as a through h. Assume that the circuit has two outputs. The grey shading, which identifies the undetected fault group g_0, indicates that all faults are undetected in the initial fault list. Also, fault coverage (FC) and diagnostic coverage (DC) are both initially 0. We assume that there are three vectors generated by a detection ATPG and can detect all faults, thus having 100% FC. Later in the simulation process more vectors will be generated by diagnostic ATPG [16] to improve diagnostic coverage (DC).

The first vector is simulated for all eight faults using a conventional fault simulator. We use a *full-response* simulator that gives us the fault detection information for each primary output (PO). Suppose we find that the first vector detects a, e and g. Also, faults a and e are detected only on the first output and g is detected on both outputs. Thus, fault pairs (a, g) and (e, g) are distinguishable, while the pair (a, e) is not distinguishable. The result is shown in the second list in Figure 1. The fault list is partitioned into three groups. The first two groups, g_1 and g_2, shown without shading contain detected faults. Group g_0 now has 5 faults. Each group contains faults that are not distinguished from others within that group, but are distinguished from those in other groups. Counting detected faults, the fault coverage is 3/8 and counting detected fault groups, the diagnostic coverage is 2/8.



g_0
FC
DC
Initial fault list
a b c d e f g h
0/8
0/8
g_1 g_2 g_0
Detection test 1detects faults a, e, g
a e g b c d f h
3/8
2/8
Drop g
g_1 g_2 g_3 g_0
Detection test 2 detects faults b, d
a e g b d c f h
5/8
3/8
g_1 g_6 g_2 g_3 g_4 g_5
Detection test 3 detects faults a, c, f, h
a e g b d c f h
8/8
6/8
Drop a, e, h
g_1 g_6 g_2 g_3 g_4 g_5
Exclusive test for pair b, d is impossible
a e g b d c f h
7/7
6/7
Eliminate d
g_1 g_6 g_2 g_3 g_4 g_7 g_5
Exclusive test 4 detects fault c
a e g b c f h
7/7
7/7

Figure 1. Illustration of diagnostic fault simulation.

Fault g, which is in a single fault group, is dropped from further simulation. Because this fault has been uniquely distinguished from all other faults, its distinguishability status will not change by other vectors. Note that pair-wise distinguishability provided by future vectors can only subdivide the groups and subdivision of a group with just one fault will be impossible. *The fact that faults can be dropped in diagnostic fault simulation is not always recognized*. However, fault dropping is possible here only because our interest is in diagnostic coverage and not in minimizing the vector set. Seven faults are now simulated for the second vector, which detects faults b and d. Suppose, b and d are detected at the same set of outputs and hence are placed within same partition g_3. Thus, $FC = 5/8$ and $DC = 3/8$. No new fault can be dropped at this stage.

Vector 3 detects faults a, c, f and h increasing the fault coverage to 100%. Suppose c and f are detected at the same set of outputs and so are placed together in group g_4. Detection at different outputs distinguishes h from these two and hence h is placed in a separate group g_5. Also, noting that this test distinguishes between a and e, group g_1 is split into g_1 and g_6. Now, $FC = 8/8 = 1.0$ and $DC = 6/8$. Faults in fault groups with single fault are dropped.

Having exhausted the detection vectors, we find that two pairs, (b, d) and (c, f), are not distinguished. We supply target fault pair (b, d) to a diagnostic ATPG system described in [16]. Suppose we find that an exclusive test, i.e., a test that detects any one fault but not the other, is impossible thus indicating that two faults are equivalent. We remove one of these faults, say d, from g_3 and from the fault list as well. This does not change fault coverage since FC = 7/7, but improves the diagnostic coverage to DC = 6/7. All faults except c and f are now dropped from further simulation.

The only remaining fault pair (c, f) is targeted and an exclusive test is found. Suppose fault f is detected by this vector but c is not detected Thus, g_4 is partitioned to create group g_7 with fault f. The new partitioning has just one fault per group, FC = 7/7, and DC = 7/7.

4 Dictionary Construction

Fault dictionary is necessary in a cause-effect diagnosis. It facilitates faster diagnosis by comparing the observed behaviors with pre-computed signatures in the dictionary [13]. One common form of dictionary is the full-response (FR) dictionary, which stores all output responses of each faults for each test. But the problem is the size of a FR dictionary can grow prohibitively large, i.e., $(F \times V \times O)$ where F is the number of faults, V is number of vectors, and O is number of primary outputs.

Much work has been done to reduce the size of the FR dictionary [4, 10, 11]. Here we assign integers to different output responses. Thus the largest integer needed to index all different syndromes in the worst case will be $minimum(2^n - 1,\ F \times V)$ where n is number of primary outputs, F is number of faults, and V is number of vectors. However, it should be noted that faults in a same logic cone tend to produce identical output respones for a given vector set, so that the largest index is usually much smaller than $F \times V$.

Faults	Output responses			
	t1	t2	t3	t4
a	10	00	10	X
b, d	00	01	00	X
c	00	00	01	00
e	10	00	00	X
f	00	00	01	11
g	11	X	X	X
h	00	00	10	X

Figure 2. FR Dictionary.

The dictionary shown in Figure 2 is generated based on the example in Figure 1. Among the entries, X means the fault is already dropped and not simulated, 0 stands for pass (same as fault-free response), and 1 stands for fail. To reduce the dictionary size we assign integers to index different output responses. In this example, "10", "11", and "01" are indexed with 1, 2, 3, as shown in Figure 3. Although for small circuits the compression is not obvious, for larger ISCAS85 benchmark circuits the reduction can be as high as an order of magnitude.

	t1	t2	t3	t4
a	1	0	1	X
b, d	0	3	0	X
c	0	0	3	0
e	1	0	0	X
f	0	0	3	2
g	2	X	X	X
h	0	0	1	X

Figure 3. Compressed Dictionary.

Because of fault dropping in our simulator there will be 'X' in the generated dictionary. This limits the use of fault dictionary to single stuck-at fault. For a real defect the faulty respones may have no match in the dictionary. To solve this problem we introduce a heuristic.

$$\frac{Hamming\ Distance}{O \times (V - X)} \leq\ threshold\ value \qquad (3)$$

Here hamming distance is calculated from observed response to the stored syndromes, ignoring 'X's. O is the number of primary outputs, V is number of vectors, and X is number of 'X's for a fault in the dictionary. If the calculated result is smaller than a given threshold, the corresponding fault will be added to a candidate list. Then fault simulation without fault dropping will be performed on this list to obtain additional information to further narrow down upon a fault candidate.

5 Results

We used the fault simulation program Hope [9] for obtaining the fault detection data. This program was modified to obtain detection information separately at each primary output. Fault grouping for diagnostic fault simulation was implemented in the Python programming language [12]. Both programs were run on a PC based on Intel Core-2 duo 2.66GHz processor with 3GB memory. Vectors were generated using a ATPG system with diagnostic test generation capability [16]. As an illustration, the results for c432 were as follows:

Number of structurally collapsed faults: 524

Number of vectors simulated: 69

Undetected faults: 4

Maximum fault coverage, FC: 99.24%

(reached at vector 51)

DC for 51 vectors: 91.985%

Number of undistinguished groups: 13

Largest size of undistinguished group: 2

Diagnostic coverage DC: 97.506%

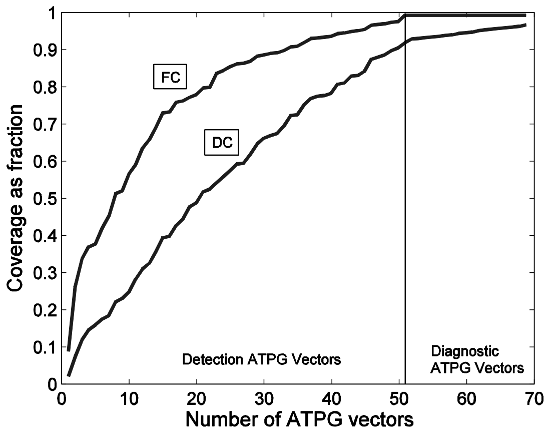


Figure 4. Diagnostic fault simulation of c432 for 69 algorithmic vectors. FC**: fault coverage,** DC**: diagnostic coverage.**

First 51 vectors detected all detectable faults; this circuit has four redundant faults. However, a fault simulator does not identify redundancies. Diagnostic fault simulation computed the diagnostic coverage of 51 vectors as 91.985%. The diagnostic coverage of all 69 vectors was 97.506%. No group had more than 2 faults.

Fault coverage (FC) and diagnostic coverage (DC) as functions of number of vectors are shown in Figure 4. We also simulated a set of 69 random vectors and their coverages are shown in Figure 5. As expected, both fault coverage and diagnostic coverage are lower than those for algorithmic vectors. Results for several ISCAS'85 circuits simulated for diagnostic coverages of deterministic vectors [16] are given in Table 1.

We draw several inferences from these results. For circuit c432, a closer examination of 13 undistinguished fault pairs showed that all are *functionally* equivalent. Updating the fault list by removing one fault from each equivalent pair will increase DC to 100% and reduce the size of the largest fault group to 1. Just as redundancy identification can give a more realistic fault coverage, sometimes referred to as *fault efficiency* [3], *functional equivalence identification* [2] can give a higher and more realistic diagnostic efficiency.

Table 1 indicates a dropping DC as circuit size increases. Notice 59.38% DC for c1355. This circuit is functionally equivalent to c499, which has a large number of XOR gates. In c1355, each XOR gate is expanded as four NAND gates. This implementation of XOR function is known to have several functionally equivalent faults. As reported [2] the structurally collapsed set of 1,574 faults reduces to 950 faults when functional collapsing is used. If we use the set of 950 faults, same 87 vectors of Table 1 will show higher DC. The advantage of functional fault collapsing, though

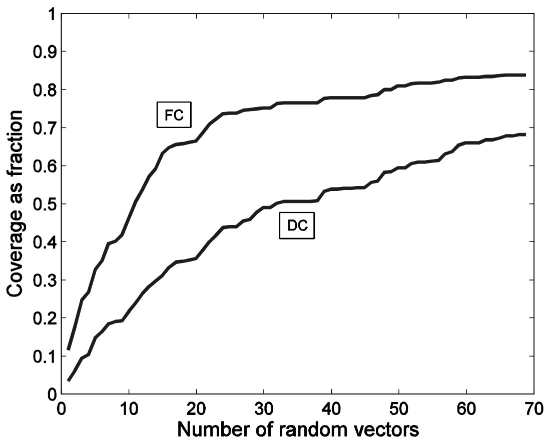


Figure 5. Diagnostic fault simulation of c432 for 69 random vectors. FC**: fault coverage,** DC**: diagnostic coverage.**

marginal in detection ATPG, can be significant in diagnostic test generation. We further notice that the size of the largest undiagnosed fault group tends to increase for larger circuits. It is 11 for c2670. This is related to the lower DC , whose reciprocal is the diagnostic resolution (DR) [1]. $DR > 1$ indicates poor diagnosis; the *ideal* resolution $DR = 1$ requires that each undistinguished fault group is no larger than 1.

In general, the time complexity of a conventional fault simulation program is linearly dependent on each of the three variables, namely, number of gates, number of faults and number of vectors. The CPU times in Table 1 include the time of the coventional fault simulator Hope [9] and that of our Python program that partitions the fault list and computes DC. The overall increase in run times with increasing circuit size for diagnostic simulation shown in Table 1 is between $O(gates^2)$ and $O(gates^3)$, which is no different from what has been reported for conventional fault simulation [3, 5]. The last column in Table 1 is the CPU time without fault dropping, which indicates a reduction of about one half to one third. With increasing circuit size the CPU tme reduction tends to increase. For simulation without fault dropping, each vector needs to be applied to all faults, thus spending approximately same amount of time, while with fault dropping a fault is immediately dropped if it is detected in detection simulation or fully diagnosed in diagnostic simulation, so that later vectors will require less and less CPU time.

6 Conclusion

The diagnostic fault simulation presented here is a core algorithm and should find effective use in the test generation systems of the future. The algorithm has similar complexity as conventional simulation with fault

Table 1 Diagnostic Fault Simulation of ISCAS'85 benchmark circuits.

Circuit	Number of faults	Number of vectors	Fault coverage FC (%)	Largest undiagnosed group size	Diagnostic coverage DC (%)	CPU s	CPU s (no fault dropping)
c17	22	8	100.0	1	100.0	0.00	0.00
c432	524	69	99.24	2	97.51	0.14	0.30
c499	758	53	100.0	2	98.40	0.13	0.31
c880	942	60	100.0	2	94.16	0.19	0.45
c1355	1574	87	100.0	3	59.38	0.70	1.63
c1908	1879	134	99.89	8	86.46	1.28	2.89
c2670	2747	150	98.84	11	86.42	2.80	6.07
c3540	3428	174	100.0	8	89.69	2.00	5.74
c6288	7744	137	99.56	3	86.87	4.12	10.23
c7552	7550	296	98.25	7	86.85	5.34	14.67

dropping. Because this fault simulation is done with fault dropping, the syndromes will contain 0, 1, and X (don't care). However, these don't cares do not reduce the diagnosability of a fault. Although, reordering or compaction of vectors will be affected. We observe that a low diagnostic coverage (DC) can result from two reasons. First, low DC of random vectors can be improved with the help of a diagnostic ATPG. Second, DC may still not be 100% due to functional equivalences that are generally not recognized in the conventional structural fault collapsing. A diagnostic fault simulator can identify fault groups that are potential targets for functional equivalence checking [15]. We can also exploit the fact that the distance between equivalent faults is generally small [7]. Thus an equivalence checking algorithm can be developed based on extracting a subcircuit that contains the fault pair. Reference [8] shows that some detection test sets have good diagnostic capability. This indicates that most of the undistinguished fault pairs might be equivalent making equivalence checking even more important.

Acknowledgment – This research is supported in part by the National Science Foundation Grant CNS-0708962.

References

[1] V. D. Agrawal, D. H. Baik, Y. C. Kim, and K. K. Saluja, "Exclusive Test and its Applications to Fault Diagnosis," in *Proc. 16th International Conf. VLSI Design*, Jan. 2003, pp. 143–148.

[2] V. D. Agrawal, A. V. S. S. Prasad, and M. V. Atre, "Fault Collapsing via Functional Dominance," in *Proc. International Test Conf.*, 2003, pp. 274–280.

[3] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory & Mixed-Signal VLSI Circuits*. Boston: Springer, 2000.

[4] B. Chess and T. Larrabee, "Creating Small Fault Dictionaries," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 346-356, Mar. 1980.

[5] P. Goel, "Test Generation Costs Analysis and Projections," in *Proc. 17th Design Automation Conf.*, 1980, pp. 77–84.

[6] T. Grüning, U. Mahlstedt, and H. Koopmeiners, "DIATEST: A Fast Diagnostic Test Pattern Generator for Combinational Circuits," in *Proc. IEEE/ACM Intl. Conf. on Computer-Aided Design*, pp. 194-197, Nov. 1991.

[7] I. Hartanto, V. Boppana, and W. K. Fuchs, "Diagnostic Fault Equivalence Identification Using Redundancy Information & Structural Analysis," in *Proc. International Test Conf.*, Oct. 1996, pp. 20-25.

[8] I. Hartanto, V. Boppana, J. H. Patel, and W. K. Fuchs, "Diagnostic Test Pattern Generation for Sequential Circuits," in *15th IEEE VLSI Test Symp.*, May 1997, pp. 196-202.

[9] H. K. Lee and D. S. Ha, "HOPE: An Efficient Parallel Fault Simulator for Synchronous Sequential Circuits," *IEEE Trans. Computer-Aided Design*, vol. 15, no. 9, pp. 1048–1058, Sept. 1996.

[10] D. Lavo and T. Larrabee, "Making Cause-Effect Cost Effective: Low-resolution Fault Dictionaries," in *Proc. International Test Conf.*, 2001, pp. 278-286.

[11] I. Pomeranz and S. M. Reddy, "On the Generation of Small Dictionaries for Fault Location," in *Proc. Intl. Conf. Computer-Aided Design*, 1992, pp. 272-278.

[12] G. van Rossum and F. L. Drake, Jr., editors, Python Tutorial Release 2.6.3. docs@python.org: Python Software Foundation, Oct. 2009.

[13] M. A. Shukoor and V. D. Agrawal, "A Two Phase Approach for Minimal Diagnostic Test Set Generation," in *Proc. 14th IEEE European Test Symp.*, May 2009, pp. 115-120.

[14] S. Venkataraman, I. Hartanto, W. K. Fuchs, E. M. Rudnick, S. Chakravarty, and J. H. Patel, "Rapid Diagnostic Fault Simulation of Stuck-at Faults in Sequential Circuits using Compact List," in *Proc. Design Automation Conf.*, pp. 133-138, June 1995.

[15] X. Yu, M. E. Amyeen, S. Venkataraman, R. Guo, and I. Pomeranz, "Concurrent Execution of Diagnostic Fault Simulation and Equivalence Identification During Diagnostic Test Generation," in *Proc. 21st IEEE VLSI Test Symp.*, May 2003, pp. 351-356.

[16] Yu Zhang, V. D. Agrawal, "A Diagnostic Test Generation System and a Coverage Metric," in *15th IEEE European Test Symp.*, May 2010, *submitted*.